WEST Search History

Hide Items | Restore | Clear | Cancel

DATE: Wednesday, August 18, 2004

Hide?	Set Name	Query	Hit Count
		DB=PGPB,USPT,USOC; PLUR=YES; OP=ADJ	
☐	L13	L11 and event	60
☐	L12	L11 and logical state	1
☐	L11	L9 and L6	77
☐	L10	L9 and reconcil$	27
☐	L9	L5 and l3	520
		DB=EPAB,JPAB,DWPI,TDBD; PLUR=YES; OP=ADJ	
☐	L8	L7 or l6	3
		DB=PGPB,USPT,USOC,EPAB,JPAB,DWPI,TDBD; PLUR=YES; OP=ADJ	
☐	L7	L5 and ISA	264
☐	L6	L5 and pipeline	803
☐	L5	(simulat$ or emulat$) and equivalen$ and verification	8326
☐	L4	(simulat$ or emulat$) and equivalen$	107911
		DB=PGPB,USPT,USOC; PLUR=YES; OP=ADJ	
☐	L3	L1 or 703/13-17,20-22,26-27.ccls.	4625
☐	L2	L1 and 703/13-17,20-22,26-27.ccls.	79
☐	L1	717/104-105,124-135.ccls.	1857

END OF SEARCH HISTORY

Terms used **simulation verification equivalence comparing ISA**

Found **38** of **140,980**

Sort results by: relevance

Display results: expanded form

Save results to a Binder

Search Tips

☐ Open results in a new window

Try an Advanced Search

Try this search in The ACM Guide

Results 1 - 20 of 38

Result page: **1** 2 next

Relevance scale

1 Compositional verification of concurrent systems using Petri-net-based condensation rules

Eric Y. T. Juan, Jeffrey J. P. Tsai, Tadao Murata

September 1998 **ACM Transactions on Programming Languages and Systems (TOPLAS)**, Volume 20 Issue 5

Full text available: pdf(578.81 KB) Additional Information: full citation, abstract, references, citings, index terms

The state-explosion problem of formal verification has obstructed its application to large-scale software systems. In this article, we introduce a set of new condensation theories: IOT-failure equivalence, IOT-state equivalence, and firing-dependence theory to cope with this problem. Our condensation theories are much weaker than current theories used for the compositional verification of Petri nets. More significantly, our new condensation theories can eliminate the interleaved behaviors ...

Keywords: Petri nets, boundedness, compositional verification, deadlock states, reachability analysis, reachability graphs, reachable markings

2 Scalable hybrid verification of complex microprocessors

Maher Mneimneh, Fadi Aloul, Chris Weaver, Saugata Chatterjee, Karem Sakallah, Todd Austin

June 2001 **Proceedings of the 38th conference on Design automation**

Full text available: pdf(150.33 KB) Additional Information: full citation, abstract, references, citings, index terms

We introduce a new verification methodology for modern micro-processors that uses a simple checker processor to validate the exe-cution of a companion high-performance processor. The checker can be viewed as an at-speed emulator that is formally verified to be compliant to an ISA specification. This verification approach en-ables the practical deployment of formal methods without impact-ing overall performance.

3 Verification of configurable processor cores

Marinés Puig-Medina, Gülbin Ezer, Pavlos Konas

June 2000 **Proceedings of the 37th conference on Design automation**

Full text available: pdf(79.05 KB) Additional Information: full citation, abstract, references

This paper presents a verification methodology for configurable processor cores. The simulation-based approach uses directed diagnostics and pseudo-random program generators both of which are tailored to specific processor instances. A configurable and

extensible test-bench serves as the framework for the verification process and offers components necessary for the complete SOC verification. Coverage analysis provides an evaluation of how well a specific design has been exercised, of the br ...

Keywords: co-simulation, configurable processor cores, coverage analysis, design verification, system-on-chip, test generation

4 Abstract interpretation of reactive systems

Dennis Dams, Rob Gerth, Orna Grumberg

March 1997 **ACM Transactions on Programming Languages and Systems (TOPLAS)**, Volume 19 Issue 2

Full text available: pdf(501.12 KB) Additional Information: full citation, references, citings, index terms, review

Keywords: abstract interpretation, formal methods, model checking, mu-calculus, reactive systems

5 Computing curricula 2001

September 2001 **Journal on Educational Resources in Computing (JERIC)**

Full text available: pdf(613.63 KB) html(2.78 KB) Additional Information: full citation, references, citings, index terms

6 High-level test generation for design verification of pipelined microprocessors

David Van Campenhout, Trevor Mudge, John P. Hayes

June 1999 **Proceedings of the 36th ACM/IEEE conference on Design automation**

Full text available: pdf(58.89 KB) Additional Information: full citation, references, citings, index terms

Keywords: design verification, high-level test generation, pipelined microprocessors, sequential test generation

7 VERILAT: verification using logic augmentation and transformations

Dhiraj K. Pradhan, Debjyoti Paul, Mitrajit Chatterjee

January 1997 **Proceedings of the 1996 IEEE/ACM international conference on Computer-aided design**

Full text available: pdf(261.91 KB) Publisher Site Additional Information: full citation, abstract, references, citings, index terms

This paper presents a new framework for formal logic verification. What is depicted here is fundamentally different from previous approaches. In earlier approaches, the circuit is either not changed during the verification process, as in OBDD or implication-based methods, or the circuit is progressively reduced during verification. Whereas in our approach, we actually enlarge the circuits by adding gates during the verification process. Specifically introduced here is a new technique that transf ...

Keywords: VERILAT, formal logic verification, implication-based methods, logic augmentation, logic testing, logic transformations

8 A scalable formal verification methodology for pipelined microprocessors
Jeremy Levitt, Kunle Olukotun
June 1996 **Proceedings of the 33rd annual conference on Design automation**
Full text available: pdf(134.62 KB) Additional Information: full citation, references, citings, index terms

9 Fast detection of communication patterns in distributed executions
Thomas Kunz, Michiel F. H. Seuren
November 1997 **Proceedings of the 1997 conference of the Centre for Advanced Studies on Collaborative research**
Full text available: pdf(4.21 MB) Additional Information: full citation, abstract, references, index terms

Understanding distributed applications is a tedious and difficult task. Visualizations based on process-time diagrams are often used to obtain a better understanding of the execution of the application. The visualization tool we use is Poet, an event tracer developed at the University of Waterloo. However, these diagrams are often very complex and do not provide the user with the desired overview of the application. In our experience, such tools display repeated occurrences of non-trivial commun ...

10 Automatic formal verification for scheduled VLIW code
Xiushan Feng, Alan J. Hu
June 2002 **ACM SIGPLAN Notices , Proceedings of the joint conference on Languages, compilers and tools for embedded systems: software and compilers for embedded systems**, Volume 37 Issue 7
Full text available: pdf(113.92 KB) Additional Information: full citation, abstract, references, index terms

VLIW processors are attractive for many embedded applications, but VLIW code scheduling, whether by hand or by compiler, is extremely challenging. In this paper, we extend previous work on automated verification of low-level software to handle the complexity of modern, aggressive VLIW designs, e.g., the exposed parallelism, pipelining, and resource constraints. We implement these ideas into a prototype tool for verifying short sequences of assembly code for TI's C62x family of VLIW DSPs, and dem ...

Keywords: DSP, VLIW, formal verification, symbolic execution, theory of equality with uninterpreted functions

11 Making abstract interpretations complete
Roberto Giacobazzi, Francesco Ranzato, Francesca Scozzari
March 2000 **Journal of the ACM (JACM)**, Volume 47 Issue 2
Full text available: pdf(445.53 KB) Additional Information: full citation, abstract, references, citings, index terms

Completeness is an ideal, although uncommon, feature of abstract interpretations, formalizing the intuition that, relatively to the properties encoded by the underlying abstract domains, there is no loss of information accumulated in abstract computations. Thus, complete abstract interpretations can be rightly understood as optimal. We deal with both pointwise completeness, involving generic semantic operations, and (least) fixpoint completeness. Completeness and fixpoint completeness are s ...

12 Testbench, verification and debugging: practical considerations: Behavioral consistency of C and verilog programs using bounded model checking
Edmund Clarke, Daniel Kroening, Karen Yorav
June 2003 **Proceedings of the 40th conference on Design automation**
Full text available: pdf(142.69 KB) Additional Information: full citation, references, index terms

Keywords: ANSI-C, equivalence checking, verilog

13 Utilizing symmetry when model-checking under fairness assumptions: an automata-theoretic approach

E. A. Emerson, A. P. Sistla

July 1997 **ACM Transactions on Programming Languages and Systems (TOPLAS)**, Volume 19 Issue 4

Full text available: pdf(322.55 KB) Additional Information: full citation, abstract, references, citings, index terms

One useful technique for combating the state explosion problem is to exploit symmetry when performing temporal logic model checking. In previous work it is shown how, using some basic notions of group theory, symmetry may be exploited for the full range of correctness properties expressible in the very expressive temporal logic CTL*. Surprisingly, while fairness properties are readily expressible in CTL*, these methods are not powerful enough to admit any amelioration of state explosion, wh ...

Keywords: abstraction, automata, model-checking, state explosion, symmetry, temporal logic

14 Efficient checker processor design

Saugata Chatterjee, Chris Weaver, Todd Austin

December 2000 **Proceedings of the 33rd annual ACM/IEEE international symposium on Microarchitecture**

Full text available: pdf(153.00 KB) ps(1.26 MB) Additional Information: full citation, references, citings, index terms

Publisher Site

15 A transaction-based approach to relational database specification

Serge Abiteboul, Victor Vianu

October 1989 **Journal of the ACM (JACM)**, Volume 36 Issue 4

Full text available: pdf(2.65 MB) Additional Information: full citation, abstract, references, citings, index terms, review

An operational approach to database specification is proposed and investigated. Valid database states are described as the states resulting from the application of admissible transactions, specified by a transactional schema. The approach is similar in spirit to the modeling of behavior by methods and encapsulation in object-oriented systems. The transactions considered are line programs consisting of insertions, deletions, and modifications, using simple selection conditio ...

16 Automatic Verification of In-Order Execution In Microprocessors with Fragmented Pipelines and Multicycle Functional Units

P. Mishra, N. Dutt, A. Nicolau, H. Tomiyama

March 2002 **Proceedings of the conference on Design, automation and test in Europe**

Full text available: pdf(163.99 KB) Additional Information: full citation, abstract

Publisher Site

As embedded systems continue to face increasingly higherperformance requirements, deeply pipelined processor ar-chitecturesare being employed to meet desired system per-

formance.System architects critically need modeling tech-niquesthat allow exploration, evaluation, customizationand validation of different processor pipeline configurations,tuned for a specific application domain. We propose a novelFinite State Machine (FSM) based modeling of pipelinedprocessors and define a set of properties th ...

17 A system for program refinement

Thomas E. Cheatham, Judy A. Townley, Glenn H. Holloway

September 1979 **Proceedings of the 4th international conference on Software engineering**

Full text available: pdf(898.12 KB) Additional Information: full citation, abstract, references, citings, index terms

The Program Development System (PDS) is a programming environment, an integrated collection of interactive tools that support the process of program definition, testing, and maintenance. The PDS is intended to aid the development of large programs, especially program families whose members must be maintained in synchrony. The system facilitates implementation by stepwise refinement, and it keeps a refinement history that allows program modifications made at a high level of abstraction to be ...

18 Algebraic approaches to nondeterminism—an overview

Michał Walicki, Sigurd Meldal

March 1997 **ACM Computing Surveys (CSUR)**, Volume 29 Issue 1

Full text available: pdf(863.23 KB) Additional Information: full citation, references, citings, index terms

19 Symbolic model checking for event-driven real-time systems

Jin Yang, Aloysius K. Mok, Farn Wang

March 1997 **ACM Transactions on Programming Languages and Systems (TOPLAS)**, Volume 19 Issue 2

Full text available: pdf(562.52 KB) Additional Information: full citation, references, citings, index terms

Keywords: binary decision diagrams

20 Efficient checking of temporal integrity constraints using bounded history encoding

Jan Chomicki

June 1995 **ACM Transactions on Database Systems (TODS)**, Volume 20 Issue 2

Full text available: pdf(2.70 MB) Additional Information: full citation, abstract, references, citings, index terms, review

We present an efficient implementation method for temporal integrity constraints formulated in Past Temporal Logic. Although the constraints can refer to past states of the database, their checking does not require that the entire database history be stored. Instead, every database state is extended with auxiliary relations that contain the historical information necessary for checking constraints. Auxiliary relations can be implemented as materialized relational views.

Keywords: active databases, database integrity, integrity constraints, real-time databases, temporal databases, temporal logic, triggers

CiteSeer Find: simulation AND verification AND equi Documents Citations

Searching for **simulation and verification and equivalence and comparing**.
Restrict to: Header Title Order by: Expected citations Hubs Usage Date Try: Google (CiteSeer) Google (Web) CSB DBLP
25 documents found. **Order: number of citations.**

The NCSU Concurrency Workbench - Cleaveland, Sims (1996) (Correct) (53 citations)
Commands. In addition to providing a system **simulation** facility, the NCSUCWB can compute a number of Workbench (NCSU-CWB) 1] supports the automatic **verification** of finite-state concurrent systems. The main calculating a number of different behavioral **equivalences** and preorders between systems and for
ftp.csc.ncsu.edu/pub/software_research/papers/rance/cav96b.ps.Z

One or more of the query terms is very common - only partial results have been returned. Try Google (CiteSeer).

Forward and Backward Simulations - Part II: Timing-Based Systems - Lynch, Vaandrager (1995) (Correct) (39 citations)
Forward and Backward **Simulations** Part II: Timing-Based Systems Nancy Lynch
www.cs.kun.nl/~fvaan/PAPERS/LynchVpartII.ps.Z

Multiway Synchronization Verified with Coupled Simulation - Parrow, Sjödin (1992) (Correct) (20 citations)
Multiway Synchronization Verified with Coupled **Simulation** Joachim Parrow and Peter Sjodin Swedish 8, 9, 10, 16]We will here concentrate on the **verification** methodology, i.e. the formulation and proof of coupled **simulations**. The coupled **simulation equivalence** is weaker than observation **equivalence** and
www.sics.se/~peter/papers/cs-mway.ps

Tight Integration of Combinational Verification Methods - Jerry Burch Vigyan (1998) (Correct) (13 citations)
and Matsunaga used BDDs. Of course, random **simulation** has been used for **verification** for a long time. Tight Integration of Combinational **Verification** Methods Jerry R. Burch Vigyan Singhal
verification is an important piece of most **equivalence** checking tools. In the recent past, many
andante.eecs.umich.edu/ecad/conferences/papers/1998/iccad98/htmfiles/sun_sgi/../../pdffiles/10b_3.pdf

A Front-End Generator for Verification Tools - Cleaveland, Madelaine, Sims (1995) (Correct) (13 citations)
preorder checking, model checking, random **simulation**, and abstraction mechanisms, for analyzing
A Front-End Generator for **Verification** Tools Rance Cleaveland y Eric Madelaine
and provide users different methods, such as **equivalence** checking, preorder checking, model checking,
www-sop.inria.fr/meije/papers/pac.tacas.ps.gz

"On the Fly" Verification of Behavioural Equivalences and.. - Fernandez, Mounier (1991) (Correct) (9 citations)
decision procedures for bisimulation and **simulation** relations between two transition systems. The
"On the Fly" **Verification** of Behavioural **Equivalences** and Preorders
"On the Fly" **Verification** of Behavioural **Equivalences** and Preorders Jean-Claude Fernandez Laurent
ftp.inrialpes.fr/pub/INRIA/projets/VASY/publications/cadp/Fernandez-Mounier-91-a.ps.Z

Specification and Verification of a Sliding Window Protocol .. - Madelaine, Vergamini (1991) (Correct) (5 citations)
with the simulator. It builds interactively a **simulation** tree, proposing menus of events at each step.
Specification and **Verification** of a Sliding Window Protocol in LOTOS Eric
of layered protocols is the requirement of **equivalence** between the service of a given layer and the
www-sop.inria.fr/meije/papers/forte91.paper.ps.gz

Efficient Scaling-Invariant Checking of Timed Bisimulation - Weise, Lenzkes (1997) (Correct) (5 citations)
we restrict our presentation to timed **simulation** (which is "half a bisimulation"All
and formal languages, program specification and **verification**, decidability, real-time systems. Abstract.
bisimulation (Mil89]are useful notions of **equivalence** for the **verification** and analysis of
www.cs.auc.dk/~cweise/myPapers/stacs97.ps.gz

Representing Boolean Functions with If-Then-Else DAGs - Karplus (1988) (Correct) (4 citations)
logic minimization, logic synthesis, and **simulation**. Various representations have been used, with

vlsi design. The main uses in cad have been in **verification**, logic minimization, logic synthesis, and
Using canonical forms makes checking for **equivalence** easier. For a strong canonical form, only one
ftp.cse.ucsc.edu/pub/tr/ucsc-crl-88-28.ps.Z

Simulation Preorder on Simple Process Algebras - Kucera, Mayr (1999) (Correct) (3 citations)
Simulation Preorder on Simple Process Algebras Antonn Ku
(e.g. counters, buffers)The task of formal **verification** is to prove that the specification and the
We consider the problem of **simulation** preorder/**equivalence** between infinite-state processes and
www.dcs.ed.ac.uk/home/mayrri/icalp99.ps

Towards an integrated proposal for Interactive.. - Palanque.. (1996) (Correct) (3 citations)
and for further processing such as **verification**, **simulation** and execution of the specification Evaluation
(TLIM and MICO) for the design, specification, **verification**, development and evaluation of Interactive
specification preserving strong bisimulation **equivalence**. However, both basic LOTOS and Petri nets with
lis.univ-tlse1.fr/~palanque/Ps/dsvis96f.ps.gz

A Tool Set for deciding Behavioral Equivalences - Fernandez, Mounier (1991) (Correct) (3 citations)
systems with respect to bisimulation or **simulation**-based **equivalence** relations. First, we recall
Mounier y Abstract This paper deals with **verification** methods based on **equivalence** relations
A Tool Set for deciding Behavioral **Equivalences** Jean-Claude Fernandez y Laurent Mounier
ftp.imag.fr/pub/VERIMAG/ALDEBARAN/Papers/Fernandez-Mounier-91-b.ps.gz

A Model for the Dynamic Semantics of VHDL for CAD Tool Optimization - Pandey (1995) (Correct) (1 citation)
Present day hardware complexity mandates prior **simulation** of designs to achieve some level of confidence
the language as against past efforts in formal **verification** that have only concentrated on proving the
that this semantics lends itself to proofs of **equivalences** between two different albeit correct
www.ececs.uc.edu/~paw/lab/theses/spandey.ps.gz

Using PO Methods for Verifying Behavioural Equivalences - Monica Lara (1995) (Correct) (1 citation)
language containment and equality, adapted) **simulation** refinement and bisimulation **equivalence**. We
global system by this independence relation for **verification** means. The allowed reduction depends on the
Using PO Methods for Verifying Behavioural **Equivalences** Monica Lara de Souza Robert de Simone INRIA
ftp-sop.inria.fr/meije/MeijeTools/forte95a.ps.gz

Attribute Grammar Applications in Prototyping LOTOS Tools - van Eijk (1991) (Correct) (1 citation)
communication tree of the behaviour, also called **simulation**)The second group of functions supports the
also describes how this is done. One aspect of **verification** is proving the **equivalence** of two
done. One aspect of **verification** is proving the **equivalence** of two specifications. The strong semantical
wwwtios.cs.utwente.nl/pubs/pve/waga.ps.gz

Formal Verification - Model Checking - Forsberg, Ringbo (1998) (Correct)
powerful enough to replace a major part of the **simulations** that are done today? 2. Is it possible to
Bjrn Forsberg &Ulf Ringbo Master Thesis Formal **Verification** -Model Checking Abstract Note: This Report
www.ele.kth.se/ESD/doc/ar98/mscThesis/ReportFormalVerification.ps.gz

Specification and Verification of Synchronous Hardware using LOTOS - He, Turner (1999) (Correct)
semi-formal because their semantics is based on **simulation** models. Other HDLs do have formal semantics,
1 Specification and **Verification** of Synchronous Hardware using LOTOS Ji He and
and reachability analysis, together with **equivalence** or preorder checking. DILL can thus exploit a
ftp.cs.stir.ac.uk/pub/staff/kjt/research/pubs/sync-lot.ps.gz

Adding Dense Time Properties to Finite--State.. - Courcoubetis Dill.. (1992) (Correct)
are extremely difficult to detect and debug by **simulation** or even by running an implementation of the
to this problem is to use automatic protocol **verification** programs to assure the correctness of the
many timed system states into finitely many **equivalence** classes. The first such effort, by Berthomieu
ftp.csi.forth.gr/tech-reports/1992/1992.TR072.Dense_Time_Finite-State_Machines_COSPAN.ps.Z

Comparing HOL, MDG and VIS: A Case Study on the Verification .. - Curzon, Tahar, Lu (Correct)
on ROBDDs and integrating **verification** with **simulation** and synthesis. The MDG system is an
Comparing HOL, MDG and VIS: A Case Study on the **Verification** of an ATM Switch Fabric Paul Curzon z
www.ece.concordia.ca/~tahar/pub/MDX_TR99.ps

First 20 documents Next 20

Try your query at: Google (CiteSeer) Google (Web) CSB DBLP